PICTURE ENCODING AND DECODING METHOD

Patent Number: JP9023422
Publication date: 1997-01-21
Inventor(s): OSAKO FUMINORI;; YASHIMA YOSHIYUKI
Applicant(s): NIPPON TELEGR & TELEPH CORP <NTT>
Requested Patent: JP9023422
Application Number: JP19950171816 19950707
Priority Number(s):
IPC Classification: H04N7/24; G06T9/00
EC Classification:
Equivalents: JP3283159B2

Abstract

PROBLEM TO BE SOLVED: To provide a method for realizing stable encoding and decoding with high quality in given operation ability in accordance with the change of the operation ability of CPU when a picture signal is encoded and decoded on a real time.
SOLUTION: A load state measuring part 106 measures the load state of CPU 105 and transmits a load state parameter 107 to an operation quantity setting part 108. The operation quantity setting part 108 sets the operation quantity of an encoding processing part 103 in accordance with the load state based on the load state parameter 107. The encoding processing part 103 encoding-processes an input picture signal 102 with a processing algorithm by which the operation quantity of an operation quantity variable element 104 becomes operation quantity which is set and executes encoding.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平9－23422

(43)公開日　平成９年(1997)１月21日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｈ０４Ｎ　7/24			Ｈ０４Ｎ　7/13	Ｚ	
Ｇ０６Ｔ　9/00			Ｇ０６Ｆ　15/66	３３０Ａ	

審査請求　未請求　請求項の数２　ＯＬ　（全 8 頁）

(21)出願番号　特願平7－171816

(22)出願日　平成７年(1995)７月７日

(71)出願人　000004226
日本電信電話株式会社
東京都新宿区西新宿三丁目19番２号

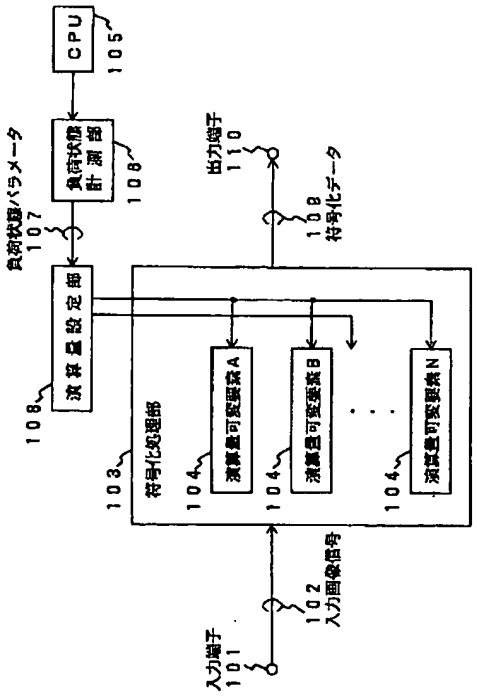
(72)発明者　大迫　史典
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内

(72)発明者　八島　由幸
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内

(74)代理人　弁理士　小笠原　吉義　（外１名）

(54)【発明の名称】　画像符号化復号化方法

(57)【要約】
【課題】画像信号を実時間で符号化復号化する場合に，ＣＰＵの演算能力の変化に対応して，与えられた演算能力の中で安定した高品質な符号化復号化を実現する方法を提供する。
【解決手段】負荷状態計測部106 は，ＣＰＵ105 の負荷状態を計測して負荷状態パラメータ107 を演算量設定部108 に送る。演算量設定部108 は，負荷状態パラメータ107 をもとに符号化処理部103 の演算量を負荷状態に応じて設定する。符号化処理部103 は，演算量可変要素104 の演算量が設定された演算量になるような処理アルゴリズムにて入力画像信号102 を符号化処理し，符号化する。



【特許請求の範囲】
【請求項１】　画像信号を符号化または復号化する際に，符号器，復号器または外付けのコンピュータのＣＰＵの負荷状態を参照し，その負荷状態を計測することによって，符号化または復号化処理に要する演算量を負荷状態に応じて適応的に変化させることを特徴とする画像符号化復号化方法。
【請求項２】　請求項１記載の画像符号化復号化方法において，符号化または復号化処理の中に演算量を変化させることができる要素が複数個存在する時に，それぞれの要素の演算量を，符号器，復号器または外付けのコンピュータのＣＰＵの負荷状態の度合によって適応的に決定することを特徴とする画像符号化復号化方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は，ソフトウェアにより計算機上で画像信号を実時間でかつ高品質に符号化・復号化することを目的とした画像符号化および復号化方法に関するものである。
【０００２】
【従来の技術】近年のコンピュータのＣＰＵ能力の伸びに対応して，ソフトウェアによって画像の符号化および復号化を実行することがしばしば行われる。ソフトウェアによる符号化復号化は，そのハンドリングの容易性から，将来が期待されている。
【０００３】ソフトウェアによる符号化の手順の例は以下のようになる。まず，ディスクあるいはテープ等に蓄積されている画像信号から，ある単位の画像データをコンピュータのメインメモリ上に呼び出す。呼び出し単位は，種々考えられるが，動画像信号の場合，１フレームごとに呼び出されることが多い。ある単位の画像データが呼び出されると，その画像データに対して，符号化処理が施され符号化データが出力される。メインメモリ上の画像データの符号化処理が終了した時点で，次の処理単位の画像データをディスクあるいはテープから読み出す。
【０００４】復号側においても，入力が符号化データとなり，出力が復号画像信号となることを除いて，符号化側と同様のプロセスで処理が実行される。このような方法によれば，動画像の呼び出し単位となっている画像データを実時間表示した場合の時間t_iと，その画像データを符号化処理するのに要する時間t_cの関係が，$t_i \geqq t_c$である場合には，実時間での処理が可能となる。たとえば，ＮＴＳＣ形式のテレビジョン信号に対しては，１／３０秒ごとに新たなフレームが入力されるが，これに対し，１フレームの符号化処理が１／３０秒以内で終了すれば良いことになる。
【０００５】しかしながら，符号化処理に際して，高画質化のために動き補償等の大きな演算量を要するアルゴリズムを用いると，ＣＰＵ等の演算処理能力の関係で，この条件が成り立たなくなるおそれがある。また，演算量が比較的少ないアルゴリズムを用いる場合でも，コンピュータ上でのソフトウェア処理の場合，他のジョブとの関係で，符号化処理に割り当てることのできる演算能力が実効的に減少し，同様にこの条件が成立しなくなる場合がある。
【０００６】従来は，符号化処理に割り当てる演算能力が減少した場合でも実時間での処理を可能とするために，あるフレームの処理が終了するまで，次々に入力されるフレームを読み飛ばし，処理が終了した時点で入力可能なフレームから符号化処理を開始するという手段で対処していた。
【０００７】図８に，従来の符号化側の処理のブロック構成を示す。入力端子８０１から入力された入力画像信号８０２は，１フレーム分の容量を持つフレームメモリ８０３に蓄積され，スイッチ８０４を通じて符号化処理部８０５に入力され，符号化データ８０６が出力端子８０７に出力される。ここで，スイッチ８０４は符号化処理部８０５からのフィードバック信号によってオン／オフされ，前フレームの画像データの符号化処理が実行されている時にはオフ，符号化処理が終了している状態ではオンになるように設定される。
【０００８】
【発明が解決しようとする課題】従来の方法によれば，動画像のフレーム間隔の時間t_iと，その画像データを符号化処理するのに要する時間t_cの関係が$t_i \geqq t_c$である場合には，スイッチ８０４は必ずオンの状態になり，すべてのフレームの処理が可能となる。
【０００９】しかしながら，$t_i < t_c$である場合には，t_cの大きさによってスイッチ８０４がオフの状態になる時間があり，その時にフレームメモリ８０３に入力されたフレームのデータは，符号化されずに捨てられることになる。たとえば，t_cがt_iの３倍である時には，３フレームに１フレームの割合でしか符号化が行われず，残りの２フレームは符号化されない。符号化されなかったフレームは受信側では符号化されたフレームの繰り返しなどで再現されるが，フレーム繰り返しで再現された画像は動きの再現性が極めて悪い。
【００１０】一方，t_cは一定ではないので，図９に示すようにＣＰＵにかかっている負荷の状態により時々刻々と変化し，符号化されるフレームの間隔もそれに応じて変化するという状況が起こりうる。
【００１１】このように，従来のような方法では，フレーム読み飛ばしにより実時間での符号化復号化処理を可能にしているために，計算機のＣＰＵに負荷がかかると急激に画質が劣化し，かつ画質の変化がＣＰＵへの負荷のかかり方の変化に非常に敏感で激しく，安定した再生画像を提供できないという問題点があった。
【００１２】
【課題を解決するための手段】本発明は，上記の問題点

を解決するためになされたものであって，ソフトウェアによって計算機上で画像信号を符号化または復号化する際に，計算機のＣＰＵの負荷状態を参照し，ＣＰＵの負荷状態の度合によって，符号化または復号化処理に要する演算量を適応的に変化させることを特徴とするものである。

【００１３】具体的には，他のジョブなどによりＣＰＵの負荷が増加している時には，符号化または復号化にかかる演算量を少なくするように処理アルゴリズムを切り替え，また逆に，ＣＰＵの負荷が減少している時には，符号化または復号化にかかる演算量を多くして，複雑な処理が可能になるように処理アルゴリズムを切り替える。

【００１４】以下，符号化側を例にとって説明する。図１に本発明を実施するための符号化側のブロック構成を示す。まず，入力端子１０１から入力された符号化対象映像の入力画像信号１０２が実時間で入力され，符号化処理部１０３へ送られる。符号化処理部１０３は，Ｎ個の演算量を変化させることのできるモジュール（演算量可変要素１０４）の集まり，すなわち，第１の演算量可変要素，第２の演算量可変要素，・・・，第Ｎの演算量可変要素１０４に分割されているとし，これらの演算を通して符号化処理が行われる。

【００１５】一方，符号化処理がなされている計算機上のＣＰＵ１０５における負荷状態計測部１０６では，そのＣＰＵ１０５にかかっているすべての負荷の状態を計測する。計測された負荷状態パラメータ１０７は演算量設定部１０８に送られ，Ｎ個の演算量可変要素に対する演算量が設定された後，Ｎ個の演算量可変要素１０４へ送られ，各演算量可変要素１０４においては，定められた演算量になるような処理アルゴリズムにて処理が行われる。処理されて得られた符号化データ１０９が，出力端子１１０に出力される。

【００１６】このような細かい演算量の制御は，従来のハードウェアによる符号化復号化処理では対処が困難であったが，ソフトウェアによる符号化復号化処理であれば，プログラム中のパラメータ変更やループの回数制限などで簡単に対応することが可能であり，実現が容易である。

【００１７】以上述べたような本発明の方法によれば，計算機のＣＰＵの負荷状態を参照しＣＰＵの負荷状態の度合を計測することにより，他のジョブなどによってＣＰＵの負荷が増加している時には，符号化または復号化にかかる演算量を少なくでき，また逆に，ＣＰＵの負荷が減少している時には，符号化または復号化にかかる演算量を多くして，複雑な処理が可能となる。また，演算量可変要素における演算量の変化は非常に細かく設定することが可能である。

【００１８】

【発明の実施の形態】以下に本発明の第１の実施の形態を示す。本実施例は，画像信号を動き補償して差分を取り，差分を量子化し可変長符号を割り当てる動き補償フレーム間符号化の符号化部を想定する。

【００１９】図２に本実施例のブロック図を示す。まず，入力端子２０１から入力された現フレームの入力画像信号２０２を１６×１６の小ブロックに分割し，分割された小ブロックに対して，フレームメモリ２０３に蓄えられている過去または未来のフレームの画像データから，動きベクトル探索部２０４において求められる動きベクトル２０５に従って最も類似するブロックを求める。求められた動きベクトル２０５に従って，動き補償部２０６において過去または未来のフレームの画像データから動き補償予測データ２０７を得る。

【００２０】現フレームの小ブロックの入力画像信号２０２と動き補償予測データ２０７を用いて，差分器２０８で動き補償フレーム間差分値２０９を算出する。動き補償フレーム間差分値２０９に対して，量子化部２１０において定められた量子化ステップ幅で量子化して量子化レベル番号を得，符号割り当て部２１１において量子化レベル番号に符号を割り当てて差分符号化データ２１２を得る。また，動きベクトル２０５にも符号を割り当てて，差分符号化データ２１２とともに多重化部２１３で多重し，符号化データ２１４として出力端子２１５から出力する。

【００２１】一方，量子化レベル番号を，逆量子化部２１６にて逆量子化して量子化代表値を得た後，動き補償予測データ２０７と加算器２１７にて加算し，その結果をフレームメモリ２０３に書き込む。

【００２２】このうち，動きベクトル探索部２０４での処理は，以下のようになっている。まず，現フレームの小ブロックＢ（ｉ，ｊ）に対して，フレームメモリ２０３中の過去または未来のフレームから空間的にＢ（ｉ，ｊ）を（ｕ，ｖ）だけずらした小ブロックＢ´（ｉ＋ｕ，ｊ＋ｖ）との差分絶対値和を計算する。

【００２３】

【数１】

$$D(u,v)=\sum_{i=0}^{15}\sum_{j=0}^{15}|B(i,j)-B'(i+u,j+v)|$$

【００２４】（ｕ，ｖ）を定められた領域で変化させ，Ｄ（ｉ，ｊ）の最小値を与えるような（ｕ，ｖ）を最終的な動きベクトル（Ｕ，Ｖ）として決定する。この動きベクトルの決定については，（ｕ，ｖ）を変化させる精度（探索精度，何画素おきに動きを探索するか）を細かくし，かつ差分絶対値和Ｄ（ｕ，ｖ）を算出する際にも１６×１６画素すべてに対して演算するようにして算出すれば，信頼性の高い動きベクトルが得られ，結果として得られるフレーム間差分値が小さくなって，符号化効率が向上することが知られている。しかしながら，この動きベクトル探索に関しては非常に演算量が多いので，

特にソフトウェアにおいて計算機上で実時間で符号化処理を行なおうとした場合に最もネックとなる部分である。

【００２５】ここで，（ｕ，ｖ）を変化させる精度と，差分絶対値和Ｄ（ｉ，ｊ）を算出する際に用いる画素数については，必ずしも全てを用いなくてもある程度の信頼性の上で動きベクトルを求めることが可能である。したがって，この２つは，演算量可変要素として扱うことが可能である。

【００２６】そこで，動きベクトル探索部２０４のうち，探索ベクトル設定部２２１と差分絶対値和計算部２２２に対して，ＣＰＵ２１８からの演算量制御を施すようにする。計算機上のＣＰＵ２１８における負荷状態計測部２１９では，そのＣＰＵ２１８にかかっているすべての負荷の状態を計測する。計測された負荷状態パラメータ２２０は演算量設定部２２３に送られ，２個の演算量可変要素，すなわち探索ベクトル設定部２２１と差分絶対値和計算部２２２に対する演算量が設定される。

【００２７】動きベクトル探索部２０４の動作を詳細に説明すれば，探索ベクトル設定部２２１において，演算量設定部２２３で定められた演算量を実現するための演算量パラメータ２２５に基づいて探索ベクトル（動きベクトルの候補）が設定され，各探索ベクトルに対して，ブロックシフト部２２４にて，フレームメモリ２０３に蓄積されている過去または未来の画像がシフトされ，差分絶対値和計算部２２２にて，現フレームの小ブロック画像データ（入力画像信号２０２）との差分絶対値和が計算される。この場合，差分絶対値和を計算するときに用いる画素数は，演算量設定部２２３で定められた演算量を実現するための演算量パラメータ２２５に基づいて定められる。

【００２８】次に，差分絶対値和が最小値検出部２２６に送られ，これにより差分絶対値和が最小になるような探索ベクトルが，動きベクトル決定部２２７にて決定され，最終的な動きベクトル２０５として求められる。

【００２９】探索ベクトル設定部２２１および差分絶対値和計算部２２２での具体的な演算量制御は，以下のような方法で行われる。計測されたＣＰＵ２１８の負荷状態パラメータ（Ｓ）２２０を，それが最大の場合を０，最小の場合を１とすると，探索ベクトル設定部２２１における探索精度Ｐは，

$P=1/S_1$

差分絶対値和計算部２２２において用いる画素数Ｑは，

$Q=256\times S_2$

と定められる。ただし，$S_1\times S_2=S$である。たとえば，Ｓ＝０．１２５の場合には，$S_1=0.25$，$S_2=0.5$とすることで実現できる。$S_1=0.25$の場合，Ｐ＝４となり，これの意味するところは，整数単位にすべてのベクトルを探索するのではなく，探索精度を１／４に減らすことである。

【００３０】すなわち，図３に探索ベクトルの演算量制御の例として示すように，探索ベクトル（動きベクトルの候補となるベクトル）を４ベクトルに１つだけにする。また，差分絶対値和を計算する画素数Ｑ＝１２８となるので，図４に示すように１画素おきに和をとっていくことになる。全体としての演算量は，$S_1\times S_2=0.125$に抑えることができる。

【００３１】種々の負荷状態パラメータ（Ｓ）２２０に対するＰ，Ｑの選択方法の例を図５に示す。このようにすれば，計算機にかかっている負荷の変動に対して非常に細かく演算量を設定するアルゴリズムを提供できる。また，本例では，演算量可変要素が２種類存在するので，請求項２の発明のように，どちらの演算量可変要素をどのくらいに設定するかを全体の負荷状態によって適応的に定めることにより，より画質の高い復号画像を提供することができる。

【００３２】図６は，本発明の第２の実施の形態を示すものである。本実施例では，画像信号を水平８画素，垂直８画素の小ブロックに分割し，小ブロックごとに離散コサイン変換して変換係数を量子化し，可変長符号を割り当てる場合を想定する。

【００３３】２次元離散コサイン変換の変換式を以下に示す。入力された小ブロックｆ（ｉ，ｊ）に対して変換係数Ｆ（ｕ，ｖ）は，

【００３４】

【数２】

$$F(u,v)=\frac{2C(u)C(v)}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}f(i,j)\cos\{\frac{(2i+1)u\pi}{2N}\}\cos\{\frac{(2j+1)v\pi}{2N}\}$$

【００３５】で表される。（Ｎ＝８）ただし，

【００３６】

【数３】

$$C(w)=\begin{cases}\frac{1}{\sqrt{2}} & w=0\\ 1 & w=1,2,3,\ldots,N-1\end{cases}$$

【００３７】である。まず，入力端子６０１から入力された現フレームの画像データである入力画像信号６０２は，離散コサイン変換部６０７へ送られる。その後，量子化部６０８へ送られ，量子化レベル番号を得て，符号割り当て部６０９において量子化レベル番号に符号を割り当てて，符号化データ６１０として出力端子６１１から出力する。

【００３８】このうち離散コサイン変換部６０７での処理は，以下のようになっている。まず，ＣＰＵ６０３の現在の負荷状態を負荷状態計測部６０４により求め，得られた負荷状態パラメータ（Ｓ）６０５に基づいて，演算量設定部６０６において演算量が設定される。設定さ

れた演算量は，離散コサイン変換部６０７に送られる。離散コサイン変換部６０７では，この設定演算量により演算量を適応的に省略する処理を行う。すなわち，負荷状態パラメータ（Ｓ）６０５が最大の場合を０，最小の場合を１としたとき，演算量可変要素として離散コサイン変換係数の計算個数Ｎを，Ｓによって以下のように切り替える。

【００３９】Ｎ＝８×８×Ｓ

このとき図７に示すように，ブロック内での周波数成分を高次の最後まで計算せずに，低次の周波数成分から順番にＮ個を計算して，ある閾値をそれ以降の高次の成分の演算は行わずに，強制的に０とする。強制的に０にした係数については，当然，引き続く量子化の制御も必要ない。このようにすれば，ＣＰＵ６０３に負荷がかかっているときほど，演算量を少なくすることができ，かつ高次の係数から演算を行うようにしているので，画質への影響の少ない符号化が可能となる。

【００４０】符号化の際の処理について説明したが，復号化の際にも以上説明した符号化処理に対応する処理が行われることは言うまでもない。その復号化処理の詳細については，以上の符号化の説明から容易に類推できるので説明を省略する。

【００４１】

【発明の効果】以上述べたような方法によれば，符号化・復号化に実効的に使用できる演算能力を演算量にフィードバックすることが可能となるので，実時間性を保持した符号化・復号化が可能となる。また，演算量可変要素を適当に選ぶことによって，実行的に使用できる演算能力の微妙な変化にも対応が可能で，急激にフレーム駒落しが起こることもないので，与えられた演算能力の中で安定した高画質な符号化を実現できる。特に，ソフトウェアによる符号化復号化処理であれば，プログラム中のパラメータ変更やループの回数制限などで簡単に実現することが可能であり，実現性も非常に高いと言える。

【図面の簡単な説明】

【図１】本発明を実施するための符号化側のブロック構成を示す図である。

【図２】本発明の第１の実施の形態のブロック構成を示す図である。

【図３】第１の実施の形態における探索ベクトルの演算量制御の例を示す図である。

【図４】第１の実施の形態における差分絶対値和計算部の演算量制御の例を示す図である。

【図５】負荷状態パラメータＳに対するＰ，Ｑの選択方法の例を示す図である。

【図６】本発明の第２の実施の形態のブロック構成を示す図である。

【図７】本発明の第２の実施の形態における離散コサイン変換の計算方法を示す図である。

【図８】従来の符号化側の処理のブロック構成を示す図である。

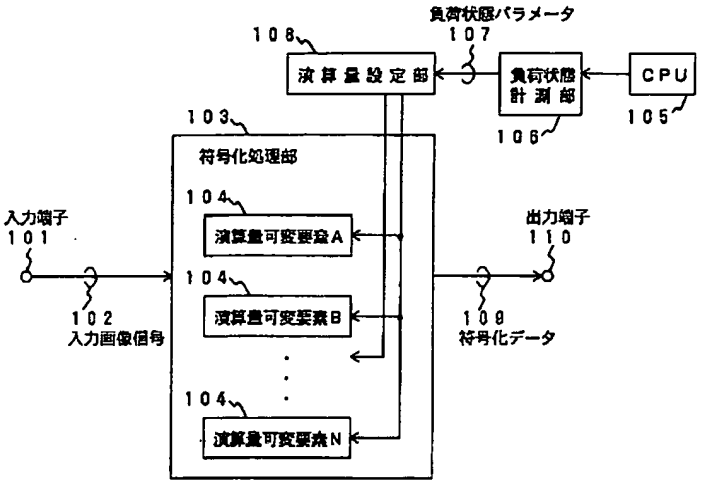
【図９】従来の符号化方法の問題点を示す図である。

【符号の説明】

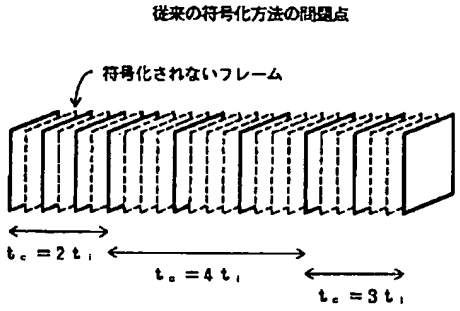
１０１　入力端子
１０２　入力画像信号
１０３　符号化処理部
１０４　演算量可変要素
１０５　ＣＰＵ
１０６　負荷状態計測部
１０７　負荷状態パラメータ
１０８　演算量設定部
１０９　符号化データ
１１０　出力端子
２０１　入力端子
２０２　入力画像信号
２０３　フレームメモリ
２０４　動きベクトル探索部
２０５　動きベクトル
２０６　動き補償部
２０７　動き補償予測データ
２０８　差分器
２０９　動き補償フレーム間差分値
２１０　量子化部
２１１　符号割り当て部
２１２　差分符号化データ
２１３　多重化部
２１４　符号化データ
２１５　出力端子
２１６　逆量子化部
２１７　加算器
２１８　ＣＰＵ
２１９　負荷状態計測部
２２０　負荷状態パラメータ
２２１　探索ベクトル設定部
２２２　差分絶対値和計算部
２２３　演算量設定部
２２４　ブロックシフト部
２２５　演算量パラメータ
２２６　最小値検出部
２２７　動きベクトル決定部
６０１　入力端子
６０２　入力画像信号
６０３　ＣＰＵ
６０４　負荷状態計測部
６０５　負荷状態パラメータ
６０６　演算量設定部
６０７　離散コサイン変換部
６０８　量子化部
６０９　符号割り当て部
６１０　符号化データ

611　出力端子
801　入力端子
802　入力画像信号
803　フレームメモリ
804　スイッチ
805　符号化処理部
806　符号化データ
807　出力端子

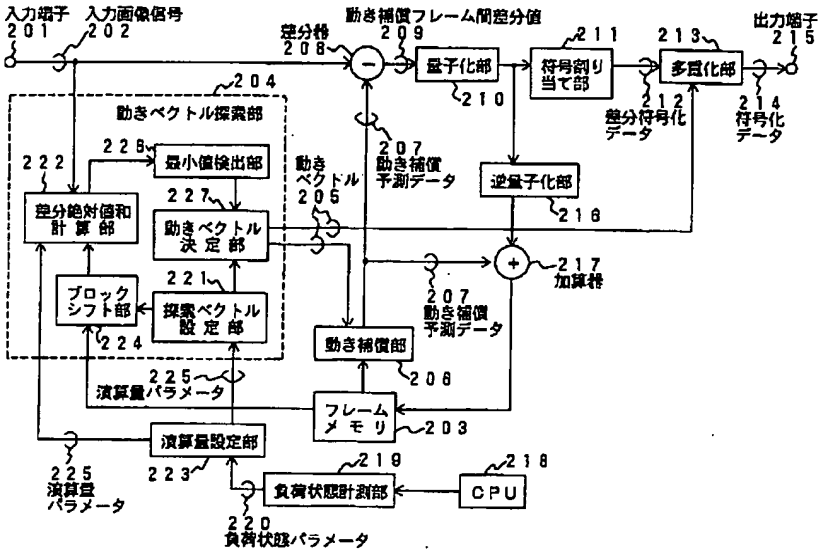
【図1】



【図9】

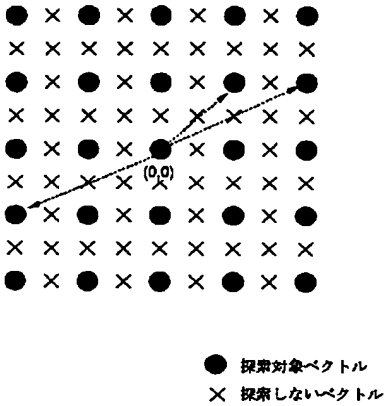


【図2】



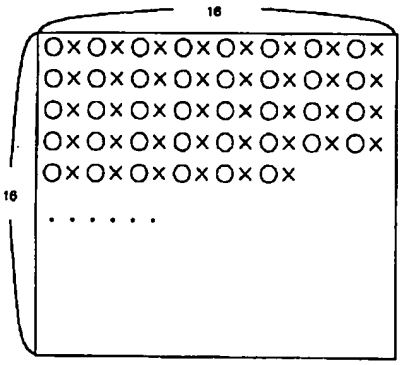
【図3】

探索ベクトルの演算量制御の例



【図4】

差分絶対値和計算部の演算量制御の例

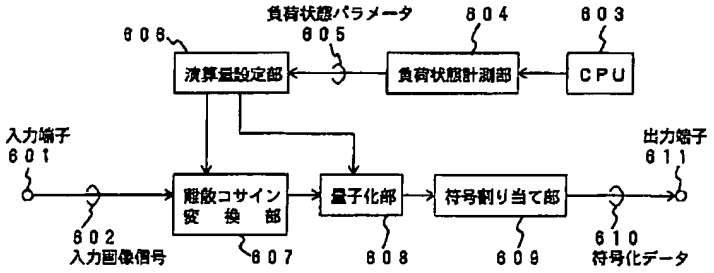


【図5】

P,Qの選択方法の例の説明図

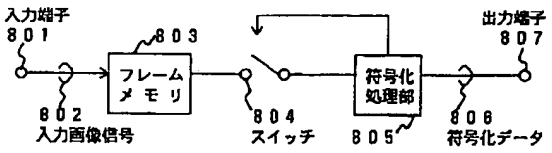
S	S_1	S_2
0.5	1	0.5
0.25	1	0.25
0.125	0.25	0.5
0.1	0.5	0.2
0.08	0.4	0.2
0.04	0.2	0.2

【図6】



【図8】

従来の符号化方法



【図7】

第2の実施例における離散コサイン変換の計算例